Assignment 5: More Scripting (Python Edition)

CS3423 - Systems Programming — Rocky Slavin - UTSA

Note: This is just a reimplementation of assignment 4, but in Python3. There are no new details or requirements besides the use of Python3 instead of bash, sed, and awk.

For this assignment, you will use **Python3** to create a simple templating engine. Your program should take as input a generic template with placeholders for generic data, a set of input files containing data which should be applied to the template, and a date. Instantiated templates using the input data will be output to a subdirectory.

This assignment requires only Python3. **Do not use** Python2, sed, awk, or bash.

Data Format

Your program should only process input data for items with an inventory of less than 10%. For example, an item with a maximum quantity of 100 and a current quantity of 9 should generate an output file, whereas an item with a maximum quantity of 50 and a current quantity of 39 should not.

1. Data files will be stored inside a directory called `data` within the same directory as your script.
2. Each file will be named based on the item number, an integer with exactly **four** digits, followed by the extension `.item`.
3. An item file consists of *exactly* three lines:
   - simple_name (string **with no whitespace**) item_name (string)
   - current_quantity (integer) max_quantity (integer)
   - body (string)
4. Example file named `3923.item`

```
b_water Bottled Water
35 99
The finest spring water you can purchase!
```

Template Format

Templates will include variable names to be filled in with data using double angle brackets. For any data file of the format described above, each of the variables (including the angle brackets) should be substituted with the data's actual value. Your program should work for **arbitrary templates** using the same variables listed below corresponding to the item values described above.

- `<<simple_name>>`
- `<<item_name>>`

- `<<current_quantity>>`
- `<<max_quantity>>`
- `<<body>>`

Example Template:

```
<html>
    <body>
        <h1><<date>> - <<item_name>></h1>
        <ul>
            <li>
                Simple Name: <<simple_name>>
            </li>
            <li>
                Quantity: <<current_quantity>>/<<max_quantity>>
            </li>
        </ul>
        <p>
            <<body>>
        </p>
    </body>
</html>
```

Output

All output files should be written to the directory defined by the last argument. Each file should be named by the item number and with the extention `.out`. For example, `3923.out`.

Date Argument

The third argument should be a date manually entered by the user of the format **MM/DD/YYYY**. This value should be substituted anywhere where `<<date>>` appears.

Script Execution

Your program should be invoked through a single Python file (see below) with **four arguments:** data directory, template file, date, and output directory. Assuming the program executes correctly, no output should print to the screen.

```
$ assign5.py ./data assign5.template 12/16/2021 ./output
```

Assignment Data

Sample input files can be found in:
`/usr/local/courses/rslavin/cs3423/Fall18/assign5`.

Script Files

Your program will consist of at least one file:

- `assign5.py` - the main file which is initially invoked

Extra Credit (5 points)

Allow your program to take *optional* fifth and sixth arguments describing the character(s) surrounding the variables instead of double angle brackets. This feature should work for the following characters as either the opening or closing symbol, /, |, }, and {. Note that these can be in any combination (e.g., starting with **{** and ending with |) If no fifth and sixth arguments are passed, the program should behave as normal. You may assume that if a fifth argument is passed, a sixth will be passed too.

Example:

```
$ assign5.py ./data assign5.template 12/16/2021 ./output '{' '|'
```

The above invocation should replace variables in the template such as `{simple_name|` instead of `<<simple_name>>`.

Extra credit is not given to late assignments. All requirements must be met to qualify for extra credit.

Submission

Turn your assignment in via Blackboard. Your zip file, named `LastnameFirstname.zip` should contain only your Python file and any related files.

If you attempt the extra credit, name your file `LastnameFirstname_EC.zip`. Without the `_EC`, your submission will be graded as normal.